Package 'valse'

May 31, 2021

Title Variable Selection with Mixture of Models

Date 2021-05-16

Version 0.1-0

Description Two methods are implemented to cluster data with finite mixture regression models. Those procedures deal with high-dimensional covariates and responses through a variable selection procedure based on the Lasso estimator. A low-rank constraint could be added, computed for the Lasso-Rank procedure. A collection of models is constructed, varying the level of sparsity and the number of clusters, and a model is selected using a model selection criterion (slope heuristic, BIC or AIC). Details of the procedure are provided in ``Model-based clustering for high-dimensional data. Application to functional data'' by Emilie Devijver (2016) <arXiv:1409.1333v2>, published in Advances in Data Analysis and Clustering.

Author Benjamin Auder <benjamin.auder@universite-paris-saclay.fr> [aut,cre], Emilie Devijver <Emilie.Devijver@kuleuven.be> [aut], Benjamin Goehry <Benjamin.Goehry@math.u-psud.fr> [ctb]

Maintainer Benjamin Auder <benjamin.auder@universite-paris-saclay.fr>

Depends R (>= 3.5.0)

Imports MASS, parallel, cowplot, ggplot2, reshape2

Suggests capushe, roxygen2

URL https://git.auder.net/?p=valse.git

License MIT + file LICENSE

RoxygenNote 7.1.1

Collate 'plot_valse.R' 'main.R' 'selectVariables.R' 'constructionModelesLassoRank.R' 'constructionModelesLassoMLE.R' 'computeGridLambda.R' 'initSmallEM.R' 'EMGrank.R' 'EMGLLF.R' 'generateXY.R' 'A_NAMESPACE.R' 'util.R'

NeedsCompilation yes

Repository CRAN

Date/Publication 2021-05-31 08:00:02 UTC

R topics documented:

`valse-package` *Variable Selection with Mixture of Models*

Description

Two methods are implemented to cluster data with finite mixture regression models. Those procedures deal with high-dimensional covariates and responses through a variable selection procedure based on the Lasso estimator. A low-rank constraint could be added, computed for the Lasso-Rank procedure. A collection of models is constructed, varying the level of sparsity and the number of clusters, and a model is selected using a model selection criterion (slope heuristic, BIC or AIC). Details of the procedure are provided in "Model-based clustering for high-dimensional data. Application to functional data" by Emilie Devijver (2016) <arXiv:1409.1333v2>, published in Advances in Data Analysis and Clustering.

Details

Two methods are implemented to cluster data with finite mixture regression models. Those procedures deal with high-dimensional covariates and responses through a variable selection procedure based on the Lasso estimator.

The main function is runValse(), which calls all other functions. See also plot_valse() which plots the relevant parameters after a run.

Author(s)

Benjamin Auder <benjamin.auder@universite-paris-saclay.fr> [aut,cre], Emilie Devijver <Emilie.Devijver@kuleuven.be> [aut], Benjamin Goehry <Benjamin.Goehry@math.u-psud.fr> [ctb]

Maintainer: Benjamin Auder <benjamin.auder@universite-paris-saclay.fr>

computeGridLambda *computeGridLambda*

Description

Construct the data-driven grid for the regularization parameters used for the Lasso estimator

Usage

```
computeGridLambda(
  phiInit,
  rhoInit,
  piInit,
  gamInit,
  X,
  Y,
  gamma,
  mini,
  maxi,
  eps,
  fast
)
```

Arguments

phiInit	value for phi
rhoInit	for rho
piInit	for pi
gamInit	value for gamma
X	matrix of covariates (of size n*p)
Y	matrix of responses (of size n*m)
gamma	power of weights in the penalty
mini	minimum number of iterations in EM algorithm
maxi	maximum number of iterations in EM algorithm
eps	threshold to stop EM algorithm
fast	boolean to enable or not the C function call

Value

the grid of regularization parameters for the Lasso estimator. The output is a vector with nonnegative values that are relevant to be considered as regularization parameter as they are equivalent to a 0 in the regression parameter.

constructionModelesLassoMLE
constructionModelesLassoMLE

Description

Construct a collection of models with the Lasso-MLE procedure.

Usage

```
constructionModelesLassoMLE(
  phiInit,
  rhoInit,
  piInit,
  gamInit,
  mini,
  maxi,
  gamma,
  X,
  Y,
  eps,
  S,
  ncores,
  fast,
  verbose
)
```

Arguments

phiInit	an initialization for phi, get by initSmallEM.R
rhoInit	an initialization for rho, get by initSmallEM.R
piInit	an initialization for pi, get by initSmallEM.R
gamInit	an initialization for gam, get by initSmallEM.R
mini	integer, minimum number of iterations in the EM algorithm, by default = 10
maxi	integer, maximum number of iterations in the EM algorithm, by default = 100
gamma	integer for the power in the penaly, by default = 1
X	matrix of covariates (of size n*p)
Y	matrix of responses (of size n*m)
eps	real, threshold to say the EM algorithm converges, by default = 1e-4
S	output of selectVariables.R
ncores	Number of cores, by default = 3
fast	TRUE to use compiled C code, FALSE for R code only
verbose	TRUE to show some execution traces

Value

a list with several models, defined by phi (the regression parameter reparametrized), rho (the covariance parameter reparametrized), pi (the proportion parameter is the mixture model), llh (the value of the loglikelihood function for this estimator on the training dataset). The list is given for several levels of sparsity, given by several regularization parameters computed automatically.

`constructionModelesLassoRank` *constructionModelesLassoRank*

Description

Construct a collection of models with the Lasso-Rank procedure.

Usage

```
constructionModelesLassoRank(
  S,
  k,
  mini,
  maxi,
  X,
  Y,
  eps,
  rank.min,
  rank.max,
  ncores,
  fast,
  verbose
)
```

Arguments

`S`	output of selectVariables.R
`k`	number of components
`mini`	integer, minimum number of iterations in the EM algorithm, by default = 10
`maxi`	integer, maximum number of iterations in the EM algorithm, by default = 100
`X`	matrix of covariates (of size n*p)
`Y`	matrix of responses (of size n*m)
`eps`	real, threshold to say the EM algorithm converges, by default = 1e-4
`rank.min`	integer, minimum rank in the low rank procedure, by default = 1
`rank.max`	integer, maximum rank in the low rank procedure, by default = 5
`ncores`	Number of cores, by default = 3
`fast`	TRUE to use compiled C code, FALSE for R code only
`verbose`	TRUE to show some execution traces

Value

a list with several models, defined by phi (the regression parameter reparametrized), rho (the covariance parameter reparametrized), pi (the proportion parameter is the mixture model), llh (the value of the loglikelihood function for this estimator on the training dataset). The list is given for several levels of sparsity, given by several regularization parameters computed automatically, and several ranks (between rank.min and rank.max).

EMGLLF *EMGLLF*

Description

Run a generalized EM algorithm developped for mixture of Gaussian regression models with variable selection by an extension of the Lasso estimator (regularization parameter lambda). Reparametrization is done to ensure invariance by homothetic transformation. It returns a collection of models, varying the number of clusters and the sparsity in the regression mean.

Usage

```
EMGLLF(
  phiInit,
  rhoInit,
  piInit,
  gamInit,
  mini,
  maxi,
  gamma,
  lambda,
  X,
  Y,
  eps,
  fast
)
```

Arguments

phiInit	an initialization for phi
rhoInit	an initialization for rho
piInit	an initialization for pi
gamInit	initialization for the a posteriori probabilities
mini	integer, minimum number of iterations in the EM algorithm, by default = 10
maxi	integer, maximum number of iterations in the EM algorithm, by default = 100
gamma	integer for the power in the penaly, by default = 1
lambda	regularization parameter in the Lasso estimation

X	matrix of covariates (of size n*p)
Y	matrix of responses (of size n*m)
eps	real, threshold to say the EM algorithm converges, by default = 1e-4
fast	boolean to enable or not the C function call

Value

A list (corresponding to the model collection) defined by (phi,rho,pi,llh,S,affec): phi : regression mean for each cluster, an array of size p*m*k rho : variance (homothetic) for each cluster, an array of size m*m*k pi : proportion for each cluster, a vector of size k llh : log likelihood with respect to the training set S : selected variables indexes, an array of size p*m*k affec : cluster affectation for each observation (of the training set)

EMGrank *EMGrank*

Description

Run an generalized EM algorithm developped for mixture of Gaussian regression models with variable selection by an extension of the low rank estimator. Reparametrization is done to ensure invariance by homothetic transformation. It returns a collection of models, varying the number of clusters and the rank of the regression mean.

Usage

```
EMGrank(Pi, Rho, mini, maxi, X, Y, eps, rank, fast)
```

Arguments

Pi	An initialization for pi
Rho	An initialization for rho, the variance parameter
mini	integer, minimum number of iterations in the EM algorithm, by default = 10
maxi	integer, maximum number of iterations in the EM algorithm, by default = 100
X	matrix of covariates (of size n*p)
Y	matrix of responses (of size n*m)
eps	real, threshold to say the EM algorithm converges, by default = 1e-4
rank	vector of possible ranks
fast	boolean to enable or not the C function call

Value

A list (corresponding to the model collection) defined by (phi,LLF): phi : regression mean for each cluster, an array of size p*m*k LLF : log likelihood with respect to the training set

`generateXY` *generateXY*

Description

Generate a sample of (X,Y) of size n

Usage

```
generateXY(n, prop, meanX, beta, covX, covY)
```

Arguments

`n`	sample size
`prop`	proportion for each cluster
`meanX`	matrix of group means for covariates (of size p)
`beta`	regression matrix, of size p*m*k
`covX`	covariance for covariates (of size p*p)
`covY`	covariance for the response vector (of size m*m)

Value

list with X (of size n*p) and Y (of size n*m)

`initSmallEM` *initSmallEM*

Description

initialization of the EM algorithm

Usage

```
initSmallEM(k, X, Y, fast)
```

Arguments

`k`	number of components
`X`	matrix of covariates (of size n*p)
`Y`	matrix of responses (of size n*m)
`fast`	boolean to enable or not the C function call

Value

a list with phiInit (the regression parameter reparametrized), rhoInit (the covariance parameter reparametrized), piInit (the proportion parameter is the mixture model), gamInit (the conditional expectation)

`plot_valse` *Plot*

Description

A function which plots relevant parameters.

Usage

```
plot_valse(X, Y, model, comp = FALSE, k1 = NA, k2 = NA)
```

Arguments

`X`	matrix of covariates (of size n*p)
`Y`	matrix of responses (of size n*m)
`model`	the model constructed by valse procedure
`comp`	TRUE to enable pairwise clusters comparison
`k1`	index of the first cluster to be compared
`k2`	index of the second cluster to be compared

Value

No return value (only plotting).

`runValse` *runValse*

Description

Main function

Usage

```
runValse(
  X,
  Y,
  procedure = "LassoMLE",
  selecMod = "DDSE",
  gamma = 1,
  mini = 10,
  maxi = 50,
  eps = 1e-04,
  kmin = 2,
  kmax = 3,
  rank.min = 1,
  rank.max = 5,
  ncores_outer = 1,
  ncores_inner = 1,
  thresh = 1e-08,
  grid_lambda = numeric(0),
  size_coll_mod = 50,
  fast = TRUE,
  verbose = FALSE,
  plot = TRUE
)
```

Arguments

X	matrix of covariates (of size n*p)
Y	matrix of responses (of size n*m)
procedure	among 'LassoMLE' or 'LassoRank'
selecMod	method to select a model among 'DDSE', 'DJump', 'BIC' or 'AIC'
gamma	integer for the power in the penaly, by default = 1
mini	integer, minimum number of iterations in the EM algorithm, by default = 10
maxi	integer, maximum number of iterations in the EM algorithm, by default = 100
eps	real, threshold to say the EM algorithm converges, by default = 1e-4
kmin	integer, minimum number of clusters, by default = 2
kmax	integer, maximum number of clusters, by default = 10
rank.min	integer, minimum rank in the low rank procedure, by default = 1
rank.max	integer, maximum rank in the low rank procedure, by default = 5
ncores_outer	Number of cores for the outer loop on k
ncores_inner	Number of cores for the inner loop on lambda
thresh	real, threshold to say a variable is relevant, by default = 1e-8
grid_lambda,	a vector with regularization parameters if known, by default numeric(0)
size_coll_mod	(Maximum) size of a collection of models, by default 50

fast	TRUE to use compiled C code, FALSE for R code only
verbose	TRUE to show some execution traces
plot	TRUE to plot the selected models after run

Value

The selected model (except if the collection of models has less than 11 models, the function returns the collection as it can not select one using Capushe)

Examples

```
n = 50; m = 10; p = 5
beta = array(0, dim=c(p,m,2))
beta[,,1] = 1
beta[,,2] = 2
data = generateXY(n, c(0.4,0.6), rep(0,p), beta, diag(0.5, p), diag(0.5, m))
X = data$X
Y = data$Y
res = runValse(X, Y, kmax = 5, plot=FALSE)
X <- matrix(runif(100), nrow=50)
Y <- matrix(runif(100), nrow=50)
res = runValse(X, Y, plot=FALSE)
```

---

selectVariables *selectVariables*

Description

For a given lambda, construct the sets of relevant variables for each cluster.

Usage

```
selectVariables(
  phiInit,
  rhoInit,
  piInit,
  gamInit,
  mini,
  maxi,
  gamma,
  glambda,
  X,
  Y,
  thresh = 1e-08,
  eps,
  ncores = 3,
  fast
)
```

Arguments

phiInit	an initial estimator for phi (size: p*m*k)
rhoInit	an initial estimator for rho (size: m*m*k)
piInit	an initial estimator for pi (size : k)
gamInit	an initial estimator for gamma
mini	minimum number of iterations in EM algorithm
maxi	maximum number of iterations in EM algorithm
gamma	power in the penalty
glambda	grid of regularization parameters
X	matrix of regressors
Y	matrix of responses
thresh	real, threshold to say a variable is relevant, by default = 1e-8
eps	threshold to say that EM algorithm has converged
ncores	Number or cores for parallel execution (1 to disable)
fast	boolean to enable or not the C function call

Value

a list, varying lambda in a grid, with selected (the indices of variables that are selected), Rho (the covariance parameter, reparametrized), Pi (the proportion parameter)

Index